```
11/19/18     (Item 10 from file: 349)
DIALOG(R)File 349:PCT Fulltext
(c) 2001 WIPO/MicroPat. All rts. reserv.

00715544    **Image available**
```

SECURE ARCHITECTURE FOR EXCHANGE EXECUTES DIGITALLY SIGNED CONTRACTS
ARCHITECTURE SECURISEE PERMETTANT L'EXECUTION D'ECHANGES DE CONTRATS A SIGNATURE NUMERIQUE

```
Patent Applicant/Assignee:
  SECURE ACCOUNTS LTD, 949 Old Ta, The Valley, AI, GB (Residence), GB
    (Nationality), (For all designated states except: US)
Patent Applicant/Inventor:
  GREEN Robert, P.O. Box 931, Shoal Bay, AI, GB (Residence), CA
    (Nationality), (Designated only for: US)
  STAMMERS Jeremy, P.O. Box 949, Old Ta, AI, GB (Residence), NZ
    (Nationality), (Designated only for: US)
  CATE Vincent, P.O. Box 949, Old Ta, AI, GB (Residence), MZ (Nationality),
    (Designated only for: US)
  HASTINGS Sean, 360 Grand Avenue #105, Oakland, CA 94610, US, US
    (Residence), US (Nationality), (Designated only for: US)
Legal Representative:
  WEISS Franklyn, Carr & Ferrell LLP, 2225 East Bayshore Road, Suite 200,
    Palo Alto, CA 94303, US
Patent and Priority Information (Country, Number, Date):
  Patent:              WO 200028452 A1 20000518 (WO 0028452)
  Application:         WO 99US25853 19991103  (PCT/WO US9925853)
  Priority Application: US 98107261 19981105; US 99292291 19990415
Designated States: AE AL AM AT AU AZ BA BB BG BR BY CA CH CN CR CU CZ DE DK
  DM EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR
  LS LT LU LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM
  TR TT TZ UA UG US UZ VN YU ZA ZW
  (EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE
  (OA) BF BJ CF CG CI CM GA GN GW ML MR NE SN TD TG
  (AP) GH GM KE LS MW SD SL SZ TZ UG ZW
  (EA) AM AZ BY KG KZ MD RU TJ TM
Main International Patent Class: G06F-017/60
International Patent Class: H04K-001/00; H04L-009/00
Publication Language: English
Filing Language: English
Fulltext Word Count: 15141
```

English Abstract

A secure architecture for value exchanges among parties (58, 60, 62), such that the parties can be widely distributed geographically and can employ a wide range of Accounting Units. It comprises a formalized legal contract in a special format, a public key that identifies each party, a computer for each party (54), a software program (56) for each party that can create, validate, sign, store, encrypt, transmit, and execute contracts, a communication method between the computers, and a database (52) for each party to hold the contracts and Identity information (98) for each party.

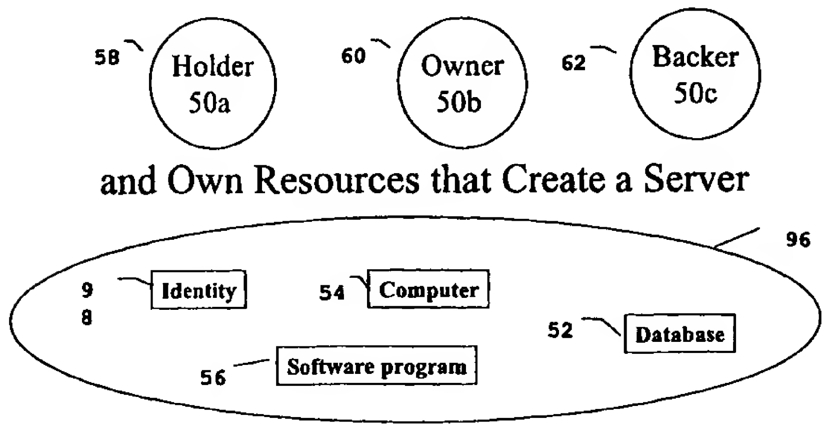
French Abstract

Architecture securisee permettant d'echanger des valeurs entre parties (58, 60, 62), de facon que lesdites parties puissent etre tres eloignees geographiquement et utiliser une grande diversite d'unites de compte. L'architecture comprend un contrat formalise conformement a la loi, presente sous un format special; une cle publique identifiant chaque partie; un ordinateur pour chaque partie (54); un programme (56) logiciel pour chaque partie, capable de creer, valider, signer, stocker, coder, transmettre et executer des contrats; une technique permettant aux ordinateurs de communiquer; et une base de donnees (52) pour chaque partie, destinee a conserver les contrats et les informations (98) d'identite de chaque partie.

Legal Status (Type, Date, Text)

Publication 20000518 A1 With international search report.
Publication 20000518 A1 Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.
Examination 20000817 Request for preliminary examination prior to end of 19th month from priority date

Parties Play Roles in Exchanges



Detailed Description

SECURE ARCHITECTURE FOR EXCHANGE EXECUTES DIGITALLY SIGNED CONTRACTS

Robert M. Green, Jeremy M. Stammers, Vincent A. Cate, Sean Hastings

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Application No. 60/107,261, filed on November 5, 1998 and entitled "SECURE ARCHITECTURE FOR EXCHANGE EXECUTES DIGITALLY SIGNED CONTRACTS". In addition, this application claims the benefit of the filing date of U.S. Application 09/292,291, filed on April 15, 1999 with the same title.

Both applications are herein incorporated by reference.

BACKGROUND

1. Field of the Invention

This invention relates to the use of computers, networks, and encryption algorithms to manage and secure exchanges of value, and more particularly to a computer architecture and system for performing and managing value exchanges that operate on a wide range of inexpensive computer systems.

2. Description of the Prior Art

Until the early 1950s, transactions to exchange value, such as invoicing, payments, wire transfers, market trades, conversions from one value unit to another, were handled manually by specialized paperwork and human processing. This manual processing was slow and error prone. As computers were applied to these activities, specific computer systems automated particular types of value transactions, usually book entry systems. Often the manual processes were copied as is into computer form and automated with the same restrictions, assumptions, and context as the manual process.

With the invention of encryption, especially public key encryption, some value transactions were made more secure from intrusion and interference. However, these prior computer systems, protocols, and architectures have the following disadvantages:

0 they are specialized for certain transactions only and often for certain geographic areas only; o some require specialized hardware, such as **Smart Cards** ; o some require private communications networks and will not work on public networks, such as SWIFT; * some require continuous, two way communication to validate and record transactions; o some are closed to new entrants (i.e., they work only with existing organizations, such as protocols for authorizing credit and-debit cards, which only work with banks; the architectures have elements of the existing organizations and businesses built into them; 0 some are closed systems which do not lend themselves to integration into customized applications; 0 they are not secure (i.e., they do not use strong encryption); 0 they cannot handle multiple accounting units in the same transactions; * they assume a hierarchical, asymmetrical relationship between parties in a value exchange (that is, someone is always the bank and who cannot also be the merchant or customer in the same transaction); in computer system terms, they distinguish strongly between the client and server - a party is either one or the other but not both; 0 they are explicitly based on a specific regulatory commissions and limited to that commissions' view of value transactions.

0 they can only handle an arbitrarily small number of parties in one transaction, often only three or four (one customer, one merchant, one bank and one credit card organization); 0 they are inherently structured to be used in a limited number of legal jurisdictions; 0 they can only function on-line and have no ability to complete an exchange off-line; 0 they assume a fixed degree of trust between parties, often specified by an organization at the top of a hierarchy that created the protocol; if you do not accept your place in the hierarchy and their rules on how much to trust parties, you cannot use the protocol; some, such as the ANSI EDI standards, are document presentation protocols, without digital signatures, non-repudiation and bookkeeping.

"Value exchanges" in the abstract sense are a part of payments, invoicing, bartering, market making, wire transfers, stock trades, issuance and clearing of checks, and currency conversions. The existing prior art performs some, but not all of these value exchanges, or only performs the communication needed prior to the exchange and not the exchange itself.
Other Efforts: There are three basic approaches to electronic payments and/or value exchanges: software-coin based, account-based (there are banks who know who the money belongs to - the present invention falls in this category), secure-hardware based (Mondex- the money is contained in the card).

0 Software-coin approach
0 eCash (tm) from Digicash.

0 Account approach
Credit Card Based Systems
o Netscape's SSL Protocol for transmitting your credit card.

o First Virtual allows you to authorize use of your credit card without giving out the actual card number.

o CyberCash(tm). Attaches a public key to a credit card.

SET Secure Electronic Transaction from
MasterCard, Visa, American Express etc
o Other Systems:
Open Trading Protocol is a method of making incompatible payment systems interoperable.

It sits on top of other payment systems such as Mondex and is supported by a consortium of payment vendors.

0 Open Financial Exchange is a standard for formatting requests and

responses in a markup language, but does not specify how the requests are processed; security via SSL is optional; JEPI - Joint Electronic Payment Initiative from CommerceNet and W3C (the World Wide Web Consortium) o The SAXASTM System (the invention described herein) Secure-Hardware approach 0 ATM Systems.

o **Smart Cards** .

o Mondex Electronic Cash. Embeds the money and the secret that protects it in a special **smart card** .

o E-Check - Electronic Checkbook on a **smart card** from the US based Financial Services Technology Consortium and FSTC members; Federal Reserve Bank, NationsBank, Bank of Boston, Huntington Bancshares, IBM and Sun Microsystems and others.
o EMV - Debit/credit cards using chip technology from Europay, MasterCard and Visa.

A number of these efforts are worth describing in more detail, in order to create a comparison between them and the present invention, the SAXAS system.

Mondex Electronic Cash extends the ATM model by making **smart cards** where the money and the secret that protects it are embedded in a special card. The security of the money is dependent upon special hardware that is difficult to break into. To spend the money, you must insert the card into an ATM or special reader. If you cannot read the internals of the card, you cannot duplicate it. If you could duplicate the card, you could steal the "money". The money belongs to the physical card; if you lose the card, you lose the money. If someone steals the card, they may not be able to spend the money, if the card has a PIN, but you have still lost it. There is no output from the card except to a card reader such as the ATM. If a phony ATM on the street was taking all your electronic money, but telling you it was not, you would have no way of knowing until you tried to use the card again later at a real ATM. Mondex will probably be used mainly for small amounts, with no accounting record.
This model of electronic money is patterned after real physical money. The value is recorded in a unique physical configuration that is difficult to counterfeit and can be used anonymously.

CyberCash, on the other hand, puts a public key around an existing credit card number in order to allow merchants to do real-time credit card authorizations. CyberCash is converting its system to use SET, which is a version of this approach created by the credit card organizations themselves.

The present invention, the SAXAS system, can easily emulate the wrapping of a credit card in a public key: in the agreement field of a SAXAS contract, you authorize another party to charge your credit card, and then you digitally sign the contract. But SAXAS can do much more:

handle multiple accounting units, match up complementary conversion orders, and transfer values without a credit card being involved.

eCashTm, a product of Digicash, offers software-based electronic coins that can be spent at eCash merchants and verified at a central point or bank. Since eCash "coins" can be digitally duplicated, the first person to the bank gets the money. They can only be used once and each has a unique serial number. There is no special hardware required, since each coin is just a string of bits that equals the right to a specific amount of money.

eCash assumes a central issuing authority, modeled after existing currency issued by governments and deposited at banks. When you transfer eCash to another party, the bank must play a part. That is because eCash coins can be duplicated and can only be spent once. Therefore, eCash must have a central database that lists all the serial numbers of coins issued and which have been spent so far. SAXAS allows accounting unit balances to move from party to party without the issuing bank being aware of the transfer. ECash uses a Blind Signature and cannot trace its own cash.

The SAXAS system can emulate the eCash capability by creating a one-time identity for a SAXAS contract, then including the private key as well as the public key within the contract. Anyone receiving the contract could sign it, execute it, then do an Owner change to move the amount from the one-time account to his or her own account.

First Virtual is an Internet payment system that allows you to circumvent sending your credit card over the Internet. You set up an account with them, register your credit card over the phone, and type in your password when prompted at sites charging using this system. You have to wait 60 days for your money, and it is one of the most barrier free methods of receiving payment over the Internet, you do not even need a merchant credit card account to accept credit cards. You do need to set up some scripts on a working First Virtual server.

Open Trading Protocol (OTP) is a consortium that is attempting to create "an open and interoperable standard for purchasing goods and services on the Internet." The model behind OTP is the existing business of purchasing goods and services. Existing types of participants in this model are explicitly identified in the model (i.e., merchant, consumer, deliverer, customer care agent, bank).

Interoperable is a key word here. This builds on top of existing payment systems to allow them to communicate. It does not actually do the payment or the exchange itself as the present invention, SAXAS, does. OTP does not assist you to create and manage your own accounting units, but if you did create one with SAXAS it might help you inter-relate it to other payment schemes. OTP derives from traditional paperwork methods of exchanging goods and services, and is a protocol for presen ting paperwork; it builds on top of and assumes some payment methods. It could build on top of SAXAS.
OTP assumes backend systems for payments and record keeping.

Account balances and inquiries for information on previous transactions are not available.

Open Financial Exchange (OFX) is a markup language for creating requests and formatting responses. It specifically tags for USA-based financial attributes such as 401K Plans, etc. Security via SSL is optional. OFX does not specify how exchanges are processed or does it perform exchanges itself.

OFX looks like an extended version of HTML (HyperText Markup Language) with tags for specific financial structures. it imposes a structure on the messages that is mapped on the specific types of financial institutions that are the intended audience of the protocol. That is, it is not a generalized method for doing any value exchange.

Instead it is a formatting standard for bank transactions, wire transfers, credit card transactions, payments, and investments, each of which is specifically identified in the protocol. For example, only certain country codes are recognized and they are listed in the protocol definition. It is hierarchical in that clients are recognized by password and servers are recognized by certificate. In SAXAS, by comparison, parties are symmetrical and the mechanism is general-purpose.

In OFX, the servers are identified by X.509 certificates. If the client and server do not share a common Certificate Authority (CA), the client cannot validate the server's certificate. Therefore, OFX specifies which CAs are to be trusted by OFX clients.

There are currently two main Certificate Authorities that have achieved market acceptance: Verisign and Thawte.

They perform due diligence on a person of firm or web page or email address, then "certify" that the public key actually came from the entit y in question, and generates an X509 certificate that a party can supply to strangers to vouch for them.

SAXAS allows all parties to identify themselves with X.509 certificates, but such certificates are optional in SAXAS and it is up to a particular SAXAS server to decide which certificates it will require. X509 certificates are not essential to performance of the invention set forth in this application. In the SAXAS system being described herein, the parties to a contract may require a particular type of certificate from the parties, or they might not, depending on the situation.

Prior Art Patents
The prior art includes Payne et al, U.S. Patent No. 5,715,314, issued Feb. 3, 1998, and discloses:

A network-based sales system includes at least one buyer computer for operation by a user desiring to buy a product, at least one merchant computer, and at least one payment computer. The buyer computer, the merchant computer, and the payment computer are interconnected

by a computer network. The buyer computer is programmed to receive a user request for purchasing a product, and to cause a payment message to be sent to the payment computer that comprises a product identifier identifying the product. The payment computer is programmed to receive the payment message, to cause an access message to be created that comprises the product identifier and an access message authenticator based on a cryptographic key, and to cause the access message to be sent to the merchant computer. The merchant computer is programmed to receive the access message, to verify the access message authenticator to ensure that the access message authenticator was created using the cryptographic key, and to cause the product to be sent to the user desiring to buy the product.

Moreau, U.S. Patent No. 5,590,196, issued Dec. 31, 1996, discloses:

Electronic funds transfer processes are being put into place to replace the paper based check clearing process. Although ubiquitous in the business environment, facsimile transmission technology has not been used for electronic transfer of funds. Fraud prevention and uncertainties in the legal status of a facsimile transmission are among the impediments for electronic funds transfer with facsimile. The method for transferring funds from a payer to a payee comprises the steps of preparing a payment form including information-for identifying an amount to be transferred, a bank of the payee and an account number of the payee, receiving and verifying a security code at an encryption unit to authorize a transmission including an encryption, preparing a facsimile transmission device to send an image of the payment form, connecting the facsimile device through the encryption unit over a communication line to a payment service provider, receiving at the payment service provider the transmission including an encryption, and sending a confirmation message to the facsimile device that the transmission has been correctly received, decrypting the encryption at the payment service provider, determining whether the encryption was authentically generated by the payer, extracting the identifying information from the facsimile transmission, and generating an electronic funds transfer request based on the identifying information provided that the encryption is determined to be authentic.

Rosen, U.S. Patent No. 5,671,280, issued Sept. 23, 1997, discloses:

A system for electronic commercial payment is provided having a customer trusted agent associated with a first money module, a merchant trusted agent that establishes a first cryptographically secure session with the customer trusted agent and associated with a second money module. Where the money modules establish a second cryptographically secure session. The customer trusted agent provides remittance advice information to the merchant trusted agent, and the merchant trusted agent provides a commercial payment ticket to the customer trusted agent. Upon receiving said commercial 12 payment ticket, the customer trusted agent initiates a transfer of electronic money from the first money module to the second money module.

Chasek, U.S. Patent No. 5,420,405, issued May 30, 1995, discloses:

This invention describes a combination of methods and apparatus that creates electronic money for personal transactions which integrates the functions of cash, checks and credit cards with constant surveilance against fraud. This money can also serve as an international medium-of-exchange, and support automated sales tax collections and payment. This money's support system is comprised of personal terminals, vendor terminals, an electronic banking sub-system, and homebase terminals. Such a system, if widely used, would increase commercial and personal productivity, provide better security against fraud and counterfeiting, facilitate the automation of operations that involve currency, and sharply diminish the flood of paper that threatens to inundate the present system.

Micali, U.S. Patent No. 5,629,982, issued May 13, 1997, discloses:

A number of electronic communications methods are described involving a first and second party (i.e., sender and recipient), with assistance from at least a trusted party, enabling electronic transactions in which the first party has a message for the second party. The first party, the second party and the 13 trusted party undertake an exchange of transmissions, such that if all transmissions reach their destinations the second party only receives the message if the first party receives at least one receipt. Preferably, the identity of the first party is temporarily withheld from the second party during the transaction. At least one receipt received to the first party enables the first party to prove the content of the message received by the second party.

Elgamal, U.S. Patent No. 5,671,279, issued Sept. 23, 1997 discloses:

A courier electronic payment system provides customers, merchants, and banks with a secure mechanism for using a public network as a platform for credit card payment services. The system governs the relationship between a Customer, Merchant, and Acquirer Gateway to perform credit card purchases over such networks as the Internet. The system uses a secure connection to simplify the problem of Internet-based financial transactions in accordance with an electronic payment protocol that secures credit card payments and certifies infrastructure that is required to enable all of the parties to participate in the electronic commerce, as well as to provide the necessary formats and interfaces between the different modules and systems.

Gifford, U.S. Patent 5,724,-424, issued Mar 3, 1998, discloses:

14
Merchant computers on the network maintain databases of digital advertisements that are accessed by buyer computers. In response to user inquiries, buyer computers retrieve and display digital advertisements from merchant computers. A digital advertisement can further include a program that is interpreted by a buyer's computer. The buyer computers include a means for a user to purchase the product described by a digital advertisement. If a user has not specified a means of payment at the time of purchase, it can be requested after a purchase transaction is initiated. A network payment system performs payment order authorization

in a network with untrusted switching, transmission, and host components. Payment orders are backed by accounts in an external financial system network, and the payment system obtains account authorizations from this external network in real-time.

Payment orders are signed with authenticators that can be based on any combination of a secret function of the payment order parameters, a single-use transaction identifier, or a specified network address.

Objects and Advantages
Accordingly, several objects and advantages of this invention are:

to be applicable to a wider range of value transactions, including invoicing, payments, purchase, sell, barter, wire transfers, market trades, conversions, exchanges, stock markets, purchase and delivery of intellectual property such as software,

time limited exchange offers, deposits, withdrawals, and loans; 0 to be secure enough that transactions are tamper proof and non-forgeable.

0 to operate on off-the-shelf, standard, inexpensive computer hardware and software as well as more expensive proprietary systems such as mainframes.

0 to use public networks such as the Internet while still maintaining privacy and security.

0 to enable any organization to create their own Accounting Units to facilitate general or limited value exchanges; examples include, but are not limited to, banks that require a way to manage customer balances, Frequent Flyer Miles for airlines, gambling tokens for casinos, cooperative marketing "coupons" that allow a customer to purchase related products from other vendors, etc.

0 to facilitate the creation of public markets in these Accounting Units by matching buyers and sellers via automatic software.
0 to integrate easily with external applications such as Internet Shopping or legacy accounting systems.

0 to be independent of any specific regulatory environment or jurisdiction and applicable to multiple jurisdictions.

0 to integrate any existing legal agreements for value exchange into a computerized system for signing, auditing, and accounting for them.

0 to maintain account balances in a way that fraud can be avoided while privacy is still maintained.

0 to be decentralized over a wide range of servers and services so that there is not a single point of failure that can bring down the entire system, either through physical failure or human intervention.

0 to have server functionality independent of a fixed network location (that is, if a server if disabled, the party running that server can reappear at another location and be recognized and in a known state relative to value transactions).
Other objects and advantages include:

to process exchanges that are not valid until a certain date and time and/or which expire on a certain date and time.

to process partial exchanges and multiple exchanges over a specified period of time at a specified limit price (example: between June 1 and June 6, buy 1000 shares of Secure Accounts Ltd. at a maximum of 100 Unibank credits per share).

to process exchanges involving unlimited numbers of parties as one exchange (i.e., either all exchanges occur or none); this invention makes it possible for a party to act as escrow agent for an arbitrarily large number of parties to a single transaction.

0 to allow competing Accounting Units and exchange services to use the same architecture 0 to allow a complete audit trail of all exchanges, enabling account balances to be regenerated if necessary due to syste msfailure or dispute.

0 to provide non-repudiatable contracts for value exchanges.
17
o to allow a selectable degree of confidence that you know who you are dealing with (the architecture allows any party providing a trusted service to require a specific type of digital signature from the parties that he deals with).

o to provide interfaces for extensibility of the architecture by other parties.

o to allow private branding of value transactions by other parties who will integrate this invention as a foundation for their specific exchange applications.

o to provide simpler methods for propagation to the parties the final transaction documents, including delivery by email or even physical mail; the underlying delivery model is neutral to the method employed - it need only be a one-way delivery.

o to allow any party to nominate any trusted third party to be the holder of account balances and **executor** of value exchanges.

o the ability to do offline transactions, such as "bank cheques", that will facilitate migration of store and forward exchange systems to fully online systems.

o to be technology neutral.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the invention, as well as other objects and further features thereof, reference may be had to the following detailed description of the invention in conjunction with the drawings wherein:

18
Fig. I shows the relationship of the parties and resources used in a value exchange; Fig. 2 shows a public key that can define and identify a party and digital signature which can verify that a party agrees to the contract; Fig. 3 shows an X.509 certificate that is an alternate way to define and identify a party; Fig. 4 shows an account defined by a unique triplet of Holder, Owner and Backer, plus a balance; Fig. 5 shows several different types of accounts; Fig. 6 shows the accounts created for one backer, two holders and three owners; Fig. 7 shows the accounts created when a holder is also an owner of units; Fig. 8 shows the structure of a value exchange contract; Fig. 9 shows the three types of contracts; Fig. 10 shows a Single-Party Owner Change Contract with an agreement; Fig. 11 shows the structure of a multi-party Owner Change contract; Fig. 12 shows Holder Change Contract, from Original Holder to Backer; Fig. 13 shows Holder Change Contract, from Backer to new Holder; Fig. 14 shows the structure of a Backer Change contract; Fig. 15 the same accounts in Fig. 6, but with Friendly Names instead of element numbers; Fig. 16 shows the execution logic for an Owner Change contract; 19 Fig. 17 shows the execution logic for a Holder Change contract; Fig. 18 shows the execution logic for a new Backer Change clause; Fig. 19a and 19b show examples of

matching Backer change clauses; Fig. 20 shows a queue of waiting Backer change clauses matched to a new backer change clause; and Fig. 21 shows the queue of Backer change clauses after matching a new clause.

SUMMARY OF THE INVENTION
This invention involves a secure architecture for value exchanges among parties, such that the parties can be widely distributed geographically and can employ a wide range of Accounting Units. It comprises a formalized legal contract in a special format, a public key that identifies each party, a computer for each party, a software program for each party that can create, validate, sign, store, encrypt, transmit, and execute contracts, a communication method between the computers, and a database for each party to hold the contracts and Identity information for each party.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT
This invention discloses an architecture for performing and managing value exchanges, with an initial embodiment in the Secure Accounts eXchange Arbitration System (also called SAXASTM) that operates on a wide range of inexpensive computer systems.

Fig. 1 shows the main components of the SAXAS architecture of value exchange. An actor in an exchange is called a Party 50. There are three roles that a party can play in any given exchange: the Holder 58, the Owner 60, and the Backer 62. A party may play more than one role in a given exchange and a party's role may change for each exchange. The architecture allows any party to be a Backer, a Holder or an owner, or all three.

In order for a party to exist and take part in exchanges, it must have a unique Identity 98, a computer 54, a database 52, and a copy of the SAXAS software program 56.

The Identity 98 of a party consists of a unique Public Key 68, Fig. 2, plus any information that the party wishes to make public. The Identity is signed by the party. This digital signature ensures that the identity information does go with the public key. The public key is also used to encrypt communications with the party, ensuring that only the intended party can read the message.

21
The information in the Identity includes a Notify Method which specifies how the SAXAS software can communicate with the party. This field will contain a URL (Universal Resource Locator), which can specify a Internet address or an email address. The combination of a Public Key and a Notify Method creates a secure channel for notifying parties of SAXAS events that are of interest to them.

The identify information may also include a preferred friendly name, telephone number, web page address, and details on any SAXAS services that the party offers.

The Identity may also include a list of attached documents, including, but not limited to, X.509 certificates and graphic logo images.

The computer 54 could comprise a personal computer (PC) running Windows@ 95 or Windows@ 98 with a minimum of 32MB of memory and at least 5MB of free disk space to manage the database, code and installation requirements.

If the computer 54 is a personal computer (PC) running Windows NT Workstation or Server software, then it requires a minimum of 64MB of memory and 5MB of free hard disk space. When the JAVA program is utilized, then versions JRE 1.2, JDK 1.1.6, JIT 1.1.7 are involved and available on the Internet at http:Hwww.javasoft.com. The database 52 components could include a database engine compatible with JDBC, such as

SQL Anywhere 5.0 from Sybase or Access from Microsoft. Also required is a JDBC driver and an ODBC drive for this database. The SAXAS software required would include an Installshield or self-extracting zip file containing the SAXAS software, JRE 1.2, JDK 1.1.6, JIT 1.1.7. The communications requirement includes a TCP/IP connection to other SAXAS sites, such as provided by an Internet Service Provider through a modem, or indirectly through a LAN using Network Interface Cards and then through an Internet Connection.

The combination of resources belonging to a party (98, 54, 52, and 56) creates a Server 96 for the Party, which is capable of managing recording the exchanges undertaken by the Party. It is possible for a single physical server to act as the logical server for a number of parties; the only requirement is that these parties share the same Transport public key for decoding incoming contracts. The function and use of these resources is explained further in the drawings and description that follow.

The Backer 62 is the party that issues or "backs" an Accounting Unit, which can also be considered as an electronic currency or virtual currency. The backer guarantees the accounting unit value by providing exchange to a specified Real World value. The backer must also act as holder and support a market for the accounting unit, matching buy and sell orders. The backer may also act as holder for other accounting units and provide exchanges to and from them to his accounting unit. The backer may also act as the owner of accounts with other holders.

The Backer can be a government that backs legal tender, a bank that backs receipts for legal tender or for shares in a Money Market Mutual fund, a,casino that issues gambling tokens, an airline that issues frequent flyer miles, a corporation that issues phone cards, or ownership shares, or coupons that can be redeemed at cooperating companies, a lawyer who issues units redeemable as hours of his time, or any other value that someone will promise to redeem under certain conditions with some real world value. The architecture does not ensure in any way that the backers are reliable or can be trusted to keep their promises. What the architecture does do is create an open market in their accounting units such that the exchange value of the accounting units can be marked down if the backer is unreliable. Therefore, what is backing an accounting unit is actually the reputation of the backer.

An Accounting Unit is equivalent to the public key that identifies the backer 62. A specific person or organization could of course have several public keys and act as backer for several Accounting Units.

The Holder 58 is typically a bank, escrow agent or broker. The Holder keeps amounts for other Owners in the Accounting Units of one or more Backers. The holder may provide various exchange services that are publicly defined in the architecture and available for computerized execution. The holder must be the owner of an account with each Backer whose Accounting Unit it holds. The holder may also be an Owner of accounts with other holders and may also be the backer of an accounting unit. The holder may provide whatever degree of anonymity or lack thereof that he decides is appropriate to his business. The Holder enforces knowledge of parties by requiring specific types of Certificates 74 in a party's Identity 98.

The Owner 60 is the signatory for one or more accounting unit balances at one or more holders. The owner must sign for any transfers from those balances. The owner may also be a holder or a backer. The owner identity may be a temporary identity that is used only one-time. The degree of anonymity allowed to an owner is determined by the holders who accept his balances.

Fig. 2 shows the format of a public key 68 and a digital signature 114. A public key is a sequence of unforgeable characters that can uniquely and

securely identify the party who holds the corresponding private key (see U.S. Patent No. 4,200,770, for the Diffie-Hellman public key cryptosystem and U.S. Patent No. 4,405,829 for RSA public key cryptosystem). In the SAXAS architecture, each party is uniquely defined by a public key. The private key is used to sign SAXAS contracts and changes to contracts that are specific to the signing party. The result is a digital signature 114, like the one shown in Fig. 2. The public key is used to verify authorizations by the party, ensuring that only the true party can authorize withdrawals of value from that account or exchanges to other Accounting Units. So, the official name and identity of a Party, whether acting as Holder, Owner or Backer, is the public key.
Fig. 3 shows another form of party identity: the X.509 digital certificate. This is a public key plus information about the real world identity of the party, all signed and verified by a certifying authority. A party who acts as a holder can require that the parties it opens accounts for have valid certificates from specified authorities. In this

way the holder can enforce the degree of anonymity it wishes to offer on accounts.

Fig. 4 shows the structure of an account 92 in the SAXAS architecture. An account is uniquely defined by the public key identities of a Holder 58, owner 60, and Backer 62, and also contains an account balance 94 in the Backer's accounting unit. The same party can appear as one, two or three of the Holder, Owner or Backer parties. Balances can be negative as in double-entry bookkeeping systems. The specific account diagrammed in Fig. 4 is interpreted as "Holder 50a owes Owner 50b the sum of 100 units of Backer 50c accounting units". Transferring a balance to a non existent account automatically creates a new account for the owner, unless this feature is turned off by the Holder. The architecture makes it easy for the Holder to charge a fee for maintaining account records and the Holder should charge enough to keep account records indefinitely.
Fig. 5 introduces the different types of accounts that can exist in the SAXAS architecture. When the Holder and the Owner are not the same party, a positive balance indicates that the Holder owes the owner the balance amount, and a negative balance indicates that the Owner owes the Holder the absolute value of the balance. When the Holder equals the Owner, the account balance indicates the total units owned by the Holder on his own behalf (this amount is zero or positive).

The following account balances from Fig. 5 are complementary and compatible.

o account 92a has a positive balance and means that Holder 50a owes 100 units of 50c to owner 50b.

o account 92b has a negative balance and means that party 50b is owed 100 units of 50c by party 50a. When account 92a exists for party 50a, then account 92b always exists for party 50b. Account 92b is held by party 50b as a claim against party 50a. Notice that the same party can be both holder and owner in different accounts.

0 account 92c records the total units issued to holder 50a by backer 50c. Since the balance is negative, the holder has a claim against the owner, who is also the backer.

o account 92d has the Holder and the owner the same party and a positive balance; this means that party 50b has a total of 100 units of 50c that are held for it by one or more holders. Account 92b shows that the 100 units are actually held by party 50a.

An account resides on the server 96, Fig. 1, belonging to the party 50 who is the Holder specified in the account triplet. Therefore, in Fig. 5, accounts 92a and 92c reside on the server of party 50a, while accounts 92b and 92d reside on the server of party 50b.

Fig. 5 also shows how accounts can be balanced against each other, both on the same server and on different servers.

Account 92a balances against 92b (that is, party 50a owes 50b 100 units, and party 50b is owed 100 units by 52a).

27
Account 92c balances against 92a (i.e., the total 50c units issued to 50a is the sum of the units held for owners such as 50b).
Fig. 6 shows the accounts that would be created to handle one backer, two holders, and three owners:

When all three parts of the triplet are the same party (Holder equals Owner equals Backer), the balance records the total units issued by the backer to all parties and is negative. In Fig. 6 account 92k shows that backer 50c has issued 50 units in total.

0 Accounts 92k-92m reside on the 50c server and record units backed by this party. Notice that accounts 921 92m balance to account 92k, which records the total obligation. Account 921 records the 20 units issued to the first holder 50a and 92m records the 30 units issued to the second holder 50b. Notice that the backer does not keep track of which party actually owns the units, only who holds them.

Accounts 92n-92p reside on the server of holder 50a and balance to zero. Account 92n is the complement of account 921 on the backer server and records the 20 units received by this holder. Accounts 92o-92p record what happened to those 20 units: 5 units are held for owner 50d and 15 units for 50e.

Accounts 92q-92s reside on the server of the second holder 50b and balance to zero. Account 92q is the 28 complement of account 92m on the backer server and records the 30 units received by this holder. Account 92r-92s record what happened to those 30 units: 20 are held for owner 50d and 10 are held for owner 50f.

Notice that party 50d has units held by both holders.

o Accounts 92t-92v reside on the server of owner 50d and balance to zero. Account 92t records the 25 units total backed by 50c that belong to this owner. Account 92u records that five of the units are held by holder 50a, and account 92v records that the other 20 are held by holder 50b. Notice that account 92v is the complement of 92r and balances it. And account 92u is the complement of 92o and balances it.

o Accounts 92w-92x reside on the server of owner 50e and balance to zero. Account 92w records the total units of 50c received by this owner and account 92x balances to account 92p on the holder server.

o Accounts 92y-92z reside on the server of owner 50f and balance to zero. Account 92y records the total units of 50c received by this owner and account 92z shows that the units are held by holder 50b (see account 92s).
Notice that the architecture can balance the sum of all the units received by owners to the total units issued by the backers, irrespective of which holder is holding them (accounts 92y, 92w and 92t total 50 units, which balances to the -50 balance of account -92k on the backer's server).

29
Fig. 7 shows how accounts are balanced when a Holder 58 is also the Owner 60 of some accounting units. In this example, backer 50c has issued 50 units to Holder 50a, of which 50a holds 10 for his own account and 40 for Owner 50b.

This is represented by the following accounts:

Accounts 92a-92b reside on the server of backer 50c and balance to zero. Account 92a records the total units that backer 50c has issued to all parties and account 92b records the units issued to Holder 50a. In this case they are the same, because only one holder is assumed. Otherwise, account 92a would balance to the sum total of the accounts recording units issued to any holder.

Accounts 92c-92e reside on the server of Holder 50a and balance to zero. Account 92c records the total units that backer 50c has issued to the holder 50a, including both those held for owner 50b and those owned by the holder himself. Account 92d records the units actually owned by the Holder 50a himself (the Holder party equals the Owner party). Account 92e records the units held on behalf of owner 50b.

Notice that account 92b on the 50c backer server also balances to account 92c on the holder server. That is because the units issued to 50c are always equal to the units received by 50c.

Accounts 92f-92g reside on the server of owner 50b and balance to zero. Account 92f records the total units backed by 50c that the owner 50b has at all holders

(only one in this case) and account 92g records the units held for 50b at holder 92a.

Notice that account 92e on the holder 50a server balances to 92g on the owner 50b server. That will always be the case when the Holder and Owner parties are reversed in the account triplet (i.e., what you have sent to be held is equal to the amount that the holder owes you).
Fig. 8 shows the structure of a value exchange contract 66. In this architecture, all exchanges of value are enacted in response to signed contracts. Each of the parties who appears as a source in the Clause section 82 must sign the top four sections (76, which comprises Header 78, Parties 80, Clauses 82, Agreement 84) of the contract in the fifth section (Signatures 86). The contracts are signed by the parties using their Identity (98, either public key 68 or digital certificate 74), Fig. 1, thus ensuring that everyone agrees to the same contract, then executed by the server.

Signatures do not oversign each other; parties sign only the signable portion of the contract 76 and the Memo, Sig-Time and Status specific to that party in the Signatures section.

Only a party listed in the party section 80 of the contract is allowed to sign in the Signatures section 86, thus allowing verification of the signature using the public key.

The contract resides in the data storage of the SAXAS server, but can be generated in various formats for external use: display (HTML-format document (HyperText Markup Language), enabling it to be rendered via a web browser), 31 edit (also HTML), transmission to another server (as a serialized object), and printing. The format that is signed is a text version of the contract (not the HTML), which the software can display for user inspection and print for submission to a legal authority or arbitrator.

Not all components of the contract are relevant to all types of contracts. The Agreement 84 is empty except in an owner change contract (see Fig. 9b). The "When Contract Becomes Valid" field of the Header 78 is only relevant to backer change contracts (see Fig. 9c); other contracts are valid as soon as they are signed by all parties. The "When Contract Expires" field is not relevant for holder change changes (see Fig. 9a) since they are complete as soon as execution starts.

Fig. 9 shows the three most common types of contracts in the architecture. There are different types of contracts with different restrictions on them in order to ensure that no account balances are ever lost or in an ambiguous state.

Since an Account 92, Fig. 4, is defined by a triplet of Holder 58, owner 60 and Backer 62, moving an amount from one account balance to another can be looked at as changing the associated Holder, Owner and/or Backer value.
Each contract specifies an **Executor** 64 in the Header section 78, Fig. 8. This party will also act as the Holder party 58 in all clauses except the destination account of Holder clauses 106. Each contract, regardless of type, may have an optional Fee Clause 90 that is the first clause in the Clause section 82. The fee clause 64 extracts a fee for executing the contract. If it exists, it does an Owner 32 Change on some amount of some accounting unit from that of the party paying the fee to that of the **Executor** . The accounting unit of the Fee Clause need not be the same as the accounting unit used in the other clauses. Therefore, there can always be one extra Backer party to the contract if the Fee clause Backer differs from the others.

There are specialized contracts in SAXAS, such as Name Service contracts with supply and request identity information, but these are all variations on the Owner Change contract, with details of the special functions stored in the agreement field.

In subsequent clauses, a contract will only modify one of the values on an account: Holder, Owner or Backer. It is theoretically possible to change more than one attribute of an account in a single contract (such as moving an amount to a new owner and changing the accounting unit at the same time), but in such cases it is more difficult to ensure that the contract executes completely or is not executed at all (i.e., that no amounts are left in limbo). Therefore, the first embodiment of SAXAS does not support that feature, in order to ensure reliability and easy auditing of the system, although other embodiments may allow this (see Alternate Embodiments).

It is the type of changes performed in the clauses section 82 that determines the type of contract and leads to the three types:

Fig. 9a: Holder change contract 98 is like a wire transfer; it moves a balance from one Holder to 33 another, perhaps to then take advantage of a conversion service offered by that Holder, but it does not change the Owner or the Backer (accounting unit). Either the source or destination Holder must be the Backer, so it takes two contracts to move from one non-Backer Holder to another non-Backer Holder. The parties to the contract will be one Owner 60, the Backer 62 and one Party who is the source or destination Holder 58. There can only be one Holder clause 106 in the Clauses section 90. The **executor** 64 will be the party originally holding the amount, not the destination party. The Agreement section 84 is empty. The Signatures section 86 contains only the signature of the Owner, the Holder and the Backer.
is
Fig. 9b: owner change contract 100 is like a check or an invoice, depending upon the author. When an invoice is signed, it becomes a check. The amount stays on the same Holder's server and in the same accounting unit, but has a new owner. Parties to the contract include at most one Holder 58 and at least one Backer 62 and at least two Owners 60, but it is permissible to include unlimited owners and backers for unlimited owner clauses 108. Either all the clauses are executed or none. The Agreement section 84 may contain one or more real world agreements that becomes part of the contract. The Agreement is open-ended and can contain multiple agreements and objects. Any digital object can be included in the Agreement by converting it into ASCII-armored format using Radix 64 (expanding groups of 3 binary 8-bit bytes into 4 printable ASCII 34 characters). The Signature section 86 must contain the signatures of all

owners who appear in a source account of any clause.

0 Fig. 9c: Backer change contract 102 is like a real world currency exchange. The server matches up the request to change Backer with other contracts which go the opposite way and that have compatible prices. The parties to the contract include exactly one holder 58, exactly two backers 62 and exactly one owner 60, although it is possible that a single party could play multiple roles in the contract. There can only be one Backer clause 110 in the Clauses section 90. The Agreement section 84 is empty. The Signatures section 86 must contain the signature of the owner 60.

By creating several contracts and executing them in sequence, it is possible to get an amount from any account to the account of any Owner residing at any Holder in the Accounting Units of any Backer (assuming that there is a market being made between the starting accounting unit and the ending accounting unit). There are additional details on the structure of these three types of contract that are shown in Fig. 10-15.

Fig. 10 shows a simple Owner Change Contract 100 in more detail. The Owner change is the simplest and most common type of contract. The Holder remains the same as does the Backer (i.e., the Accounting Unit). In this example, there is only one Holder 58 who is also the **Executor** 64 and

one Backer 62, since the fee is being paid in the same Accounting Unit as the transfer.

However, there are two Owners 60a and 60b; otherwise there could be no change of Owner. All accounts must be on the same Holder server, which must also be the **Executor** .
The only clauses allowed in the Clause section are the Fee Clause 90 and the Owner Clauses 108. The Fee Clause transfers 0.50 units of Backer 62 to the **executor** 58 in return for attempting to execute the signed contract. The first owner clause transfers 20.00 units of unit 62 to owner 60b; the MinRate of 1.00 means that the entire amount must be transferred and none can be taken by the **executor** as a fee (for this contract, the fee is corrected by the 2.00 units in the Fee clause). The second owner clause is a zero-value clause, included so that the second owner 60b will appear as a debited account and will need to sign the contract before it is valid. The second signature ensures that 60b agrees with the Agreement section that follows.

The Agreement section spells out what owner 60b agrees to do for 60a in return for the 20 units of 62. The Signature section must include signatures by 60a and 60b, since both appear as owners in the source portion of clauses.

Fig. 11 shows a more complex multi-party Owner change contract 100. Although there can only be one Holder/**Executor** in an Owner Change contract, the number of owners, backers and Owner clauses is unlimited. This makes it possible to create multi-party swaps. None of the owner clauses will be 36 executed unless all of them are executed, i.e., if one party does not have enough units to complete their part in the exchange, none of the exchanges occur.

In this example there are three owners and three accounting units (backers). For purposes of example, the three accounting units are called by the common names Lumberbucks, SoftFrancs and JackpotChips and the three owners are called Bill, Sam and June. The contract says that Bill gives Sam 200 Lumberbucks, Sam gives June 500 SoftFrancs and June gives Bill 175 JackpotChips. The **executor** /holder 58 takes 2% of each transfer as a fee for holding the balances and executing the trade. All amounts must be moved to the **Executor** 's server before the trade is initiated. And all three owners must sign the contract before it is executable.

If Fig. 11 were modified to be a two-party Owner change instead of three

parties, it would then be similar to a Backer change 102. The differences are that in an explicit Backer change the Owner does not know who provides the other accounting units and may specify a limit price as opposed to a fixed price. So a two-party Owner change contract is like an off-market, non-anonymous block trade and the Backer change contract is like an automatic buy or sell order with a range of amounts and prices that are possible to complete the order.
When a contract is being authored, signed and executed, it is necessary to know the Identities of each party to the contract. This is necessary for at least three reasons:

37
1. to know whether we wish to deal with the party (are they certified by someone we trust), 2. to know how to communicate with the party (using their Public Key and Notify Method, and 3. to learn if a party offers the service of executing a particular type of contract (i.e., conversion from one specific accounting unit to another).

The party authoring a contract agrees to provide Identity information about all the parties to the contract to all other parties to the contract. The author could obtain the Identity information from web pages, email messages, floppy disks, or any other medium. The method by which a party passes on Identity information to other parties is known as the SAXAS Name Service.

The SAXAS Name Service is implemented as Owner Change Contracts. Any party can and will act as a SAXAS Name Service at various times. The party needing the Identity information authors a contract requesting Name Service on a specific Public Key; these details are stored in the agreement section of the contract. The contract is sent to the Name Service party for execution. The Name Service inserts the desired Identity into the memo field and signs it. Any party can also pass their own Identity on to other parties by including it in their memo section of any contract and signing it.

Fig. 12 shows a Holder- Change Contract 98 in more detail. There are two basic formats: from a Holder to the Backer, or vice versa. This is an example of the first 38 format: move 400 units backed by party 62 and owned by party 60 from holder 58 to backer 62. A fee clause is used to collect 1.15 units of 62 for performing the transfer. The Backer is always the **Executor** of holder change contracts and always receives the fee, if any. The Agreement section is empty. The Signature section requires the signature of the Holder 58, Owner 60 and Backer 62.

Fig 13 shows a Holder change contract 98 that goes from the backer to a holder. In this example, the Backer 62 is acting as the original Holder as well. The amount is moved to the party 58 as the new Holder. The Fee clause allows the **executor** , who is the backer in this case, to take up to 0.75 units and move it to the account described as 62-62-62 in HOB format (this is account 92k on Fig. 6, the same account that usually has a negative total for all the units issued by this backer. So in effect the backer is withdrawing some units from circulation by collecting this fee.

The holder clause transfers 400 units from the backer's server 62 to the server of party 58, making 58 the new holder. There can only be one holder clause per holder contract. This clause also allows the **executor** to take up to 0.2% of the amount as a fee. When a Holder to Holder change is executed as two contracts and uses the Backer as an intermediary, there is no information left on the original Holder as to the final destination of the amount. Only the owner and the Backer know that the funds moved from Holder A to Holder B.

Fig. 14 shows the Backer Change Contract 102 in more 39 detail. In this case, the Fee clause is in a different accounting unit from either of the units involved in the Backer Clause itself. This is permissible under the

SAXAS architecture. The Backer clause 110 offers to give up 400 units backed by 60b in return for some units backed by 60c as long as the conversion MinRate is 0.552 or better. When the actual conversion is performed, the owner 60 will not be aware of who traded him the units. The **executor** matches up this contract with complementary contracts that offer to go the other way and also match in price. The actual conversion rate will be based on the offers available and a fee charged by the **executor** for this clause.

SAXAS allows for a single conversion clause to remain open and be matched up with multiple complimentary conversion clauses created by other owners. The clause will remain active and available for matching until the full amount offered for conversion has been converted, or the owner is found to have insufficient funds in his account, or the expiration date occurs. The details of each partial conversion are stored in the **executor** 's memo field of the contract, and re-signed by the executed after each partial conversion. See Fig. 18-21 for operational details.

Operation of the SAXAS System. Figures 15-21

Each SAXAS account is uniquely defined by an identity triplet: Holder, Owner and Backer. It is possible for each component of the triplet to be a different party, the same party, or a mixture of parties.

The Holder is the party where the account resides. The Owner is the party who signs for the units and controls the account. And the Backer is the party who defines the accounting unit for the account and "backs it". Every party in the system has a SAXAS account server and thus has accounts where it is the "holder", although these may exist only to balance with accounts that the party has with other parties.

Contracts all have an "**Executor** " that signs that a contract has been executed so that other parties to the contract know to update their books on notification. The **executor** is the final authority, or "commit coordinator".

In the case of owner change or backer change contracts it is the holder of the account. In the case of holder change contracts it is the backer of the currency, or central bank for that currency.

The **executor** does not execute a contract unless all owners of accounts giving up something (and an owner change can have more than one) have signed. Also, in a holder change the holders have to sign. The holder where the units are going to has to agree to hold the units. The holder where the units are coming from has to agree that the owner really had that much.

Fig. 15 shows the same sample set of accounts as in Fig. 6, but in this case the element numbers have been replaced with friendly names to make the logic easier to follow. Assume that the friendly name of the party backing the accounting unit is 41 $-Backer, the names of the parties acting as holders are H Bank and Union Bank, and the names of the parties acting as owners are Bob, Alice and Ted. Each Party has some Accounts on their server and some accounts on other servers, as shown in Fig. 6. It is noted that any accounts residing on a party's server will have that party as the "Holder" of the account, even though that party is acting as an owner in the overall SAXAS scheme.

Owner Change.

Fig. 16 shows the execution of an owner change contract. See Fig. 10-11 for the structure of an owner change.

Processing is simplified because the owners must move all amounts to be swapped onto the **Executor** 's server. No other servers are communicated with in the execution of a contract, although the server does send a notification copy of the completed contract to each party so that they can update their book keeping.

The optional fee clause is handled separately from the other owner change clauses. It is executed as soon as the **executor** receives a copy that is signed by the author and the fee payor, regardless of whether all the clauses of the contract turn out to be executable or not.
The **executor** verifies that any parties appearing as "debited owners" in the contract have properly signed the contract. Execution of an Owner Change is done as a unit, regardless of how many clauses and owners are involved. The 42 **Executor** logically executes all clauses simultaneously. This is done by locking all the balances in the database, executing all the transfers, then canceling the entire operation if any of the accounts falls below the required Minimum Balance (usually 0.00 or the largest negative number).

This execution model means that you can exchange something in an Owner change that you do not 'own at the start of the execution, but acquire during execution of the clauses. The **Executor** computes any percentage fees to be charged on each clause, but does not actually credit them until the end. Otherwise the **Executor** would have to lock the **Executor** 's account for the entire execution time of the contract and this would make the process inherently single threaded, since the **Executor** is involved in every contract on the server.

As a final step, the **Executor** attempts to send a notification copy of the contract as executed to each party.

This may involve use of the SAXAS Name Information Service to discover where the parties are currently to be notified.

Notification can be via email if a party is not currently online, or notification may be optional.

Returning to Fig. 15 with the friendly names, assume that there is a SAXAS Owner Change contract (as in Fig. 10 and 16) with a clause transferring $3 from Bob to Alice, the contract executing on the H-Bank server where they both have accounts. Since each account is an (HOB) "triplet", the overall transfer as executed by the H-Bank server can be summarized as 43 (H-Bank,Bob,$) $ 3 (H-Bank,Alice,$) So on the **executor** server (H-Bank) there are 2 accounts updated:

(H-Bank,Bob,$) -$3 H-Bank is holding 3 less for Bob (H-Bank,Alice,$) +$3 H-Bank is holding 3 more for Alice However, to keep the entire SAXAS system in balance, there are other accounts that must be changed as well.

Starting with the H-Bank server, when it executes the contract, it updates the two accounts mentioned above. Then it sends a notification copy of the contract to Bob and to Alice, whose SAXAS servers must also update accounts that they hold:
On Bob's Server:

(Bob,Bob,$) $25 - $3 = $22 Total $ held by Bob anywhere (Bob,H-Bank,$) -$5 + $3 = -$2 $ held by H-Bank for Bob On Alice's Server:

(Alice,Alice,$) $15+$3 $18 Total $ held by Alice anywhere (Alice,H-Bank,$) -$15-$3 -$18 $ held by H-Bank for Alice Holder Change Fig. 17 shows execution of a Holder Change Contract.

See also Fig. 12-13 for the structure of Holder Changes.

44
A Holder Change Contract is like a wire transfer in the traditional world of financial transactions - an amount is moved from one Holder to another, but retains the same Owner and Accounting Unit (i.e. Backer).

The receiving holder has to agree (sign) to hold the units before the contract is executed. This allows a holder control over whom it holds units for by automatically verifying that the party's Identity contains certain "certificates". It also makes the holder promise that he is just holding the units and that it is really still the owners.
An honest holder will have enough units on record with the Backer to be able to cover all of the owners he is holding for. When units leaves one holder the Backer for that currency will refuse a contract if the holder does not have the units to cover it. A backer is like a central bank for clearing inter-holder transfers in his currency. And acting as the central bank for his currency, the backer is the **executor** for all holder changes in his currency.

There are three cases of holder change, although the first two are a necessary and sufficient core. The third is an alternate embodiment of the method.

1) You can transfer from a holder to the backer 2) From the backer to some holder 3) From one holder to another where neither is the backer 45
Case 1: The sequence of events when going from a holder to the backer is:

1) owner signs holder change contract
2) Backer signs indicating "agree to hold for this owner" 3) The old holder A) Verifies that the owner has the units B) Updates his books and signs (he expects Backer to execute) C) Sends copy to both the backer and the owner 4) The backer executes the contract, updating his books, and change the contract status to "executed") Backer notifies owner and old holder that contract is executed

Case2: The sequence of events going from the Backer to another holder is:

1) Owner signs holder change contract
2) Destination holder signs indicating "agree to hold for this owner" 3) The backer signs and executes after checking the balance 4) The backer notifies owner and new holder that the contract is "executed" and they then update their books Case 3: Moving from holder A to holder B where neither is the backer (see Alternate Embodiments later in this document).

If an owner were moving units from one server to another, there is the danger that the first server had updated his books, but the new server did not receive the 46 notification and thus had not credited the owner with the units in his books. To correct this invalid situation, the owner would send the new holder another copy of the contract; if the contract is valid and has not been processed on the destination server yet, the server will update its books to credit the owner with units.

The Holder Change contract allows the owner to move value amounts from one Holde r to another, but without changing the Accounting Unit (Backer). The contract is signed by the Owner, the Holder and the Backer, to ensure that the amount will not be rejected when it arrives at its destination. The author will create two contracts, one to move the amount from the current holder to the backer and the second to move it on to the new holder. The reason for this is that it makes each contract atomic and unambiguous.

The owner can always verify where the funds are - they are never in limbo. If the funds get to the backer, but cannot be transferred to the new holder, they are still in the name of the owner on the backer's server and the owner can verify their status.

An Example with 2 Holder Change Contracts

Returning to Fig. 16 with the friendly names, assume that two SAXAS holder change contracts are created, with the first transferring the amount to the Backer and the second transferring it to the final destination Holder. In both cases owners creates the contracts and the

$-backer is the "**executor** ".

47
Going back to the original state of accounts in Fig.
6a, assume that Bob wants to move $5 from Union Bank to H Bank. That can be represented in (HOB) triplet notation as these two transfers:

Contract 1: (Union,Bob,$) $5 ($,Bob,$)
Contract 2: ($,Bob,$) $5 4 (H-Bank,Bob,$) Contract 1: Holder to Backer

This "(Union,Bob,$) $5 -> ($,Bob,$) " contract is NOT executed on the Union Bank server because it will be executed on the $-Backer server. However, Union Bank must both ensure that Bob has sufficient funds and ensure that he does not spend those funds once the transaction is committed. The S-Backer server must verify that Union has enough total funds to perform the transfer (although the Backer does not know if Bob has enough funds). So both servers have to sign off on sufficient funds.

The way this is handled in SAXAS is that the original holder "tentatively transfers" the funds out of Bob's account and into the Backer's account in anticipation of the contract execution. This is a reasonable expectation because the **executor** -Backer has already signed the contract saying that he will perform it once the holder signs it. If for some reason the contract does NOT go through to execution, the holder will received as negative notification from the **executor** and must be prepared to move the funds back to Bob's account.
Tentative Transfers On the Union Server:

48
(Union,$,$) -$30 + $5 = -$25 Total $ held by Union Bank (Union,Bob$) $20 - $5 = $15 $ held for Bob by Union Bank

Then the Union Bank server sends a notification copy to $-Backer and to Bob.

On the $-Backer Server:

($,Bob,$) $0 + $5 = $5 Open account for Bob, deposit $5 ($,Union,$) $30 - $5 = $25 Reduce $ holdings of Union Bank On Bob's Server:

(Bob,Union,$) -$20 + $5 = -$15 Reduce $ held at Union Bank (Bob,$,$) $0 - $5 = -$5 $-Backer now owes Bob $5 The **executor** also sends a notification copy to Union, the original holder. However, since it has already does the transfers in anticipation of this notification, it merely marks the contract as complete.

Contract 2: Backer to Holder
The $5 is now at the Backer's server. A second contract is executed to transfer it to H-Bank, still in the name of Owner Bob: " ($,Bob,$) $ 5 -> (H-Bank, Bob,$) 11

On the $-Backer Server:

($,Bob,$) $5 - $5 $0 Close out Bob, withdraw $5 49 ($,H-Bank,$) $20 + $5 = $25 Increase $ holdings of H Bank Then $-Backer sends notification copies of the signed contract to H-Bank and to the Owner Bob, so that there book entries can be updated as well. Here are the transfers on each server:

On H-Banks Server:

(H-Bank,$,$) -$20 - $5 -$25 Total $ held by H-Bank (H-Bank,Bob,$) $5 + $5 $10 Bob $ at H-Bank up by $5 On Bob's Server:

(Bob,H-Bank,$) -$5 - $5 = -$10 Increase $ owed Bob by H Bank (Bob,$,$) -$5 + $5 = $0 Close out temp account at $ Backer

Backer Change
The Backer Change Contract is a conversion of accounting units. The Owner and the Holder remain the same.
Fig. 18 shows execution of a Backer Change clause in a Backer Change Contract (see Fig. 14 for the structure). As with other contracts, the fee clause is executed at once when the contract is recorded by the **Executor** . There may also be percentage fees extracted from each conversion. The **Executor** can be any holder that makes a market. The conversion fees charged by that **Executor** are described in their identity record.

50
Each backer change clause of the contract is executed independently. Although the Backer change contract as described statically in Fig. 15 appears to involve only the Holder and the Owner. In fact the Holder matches the contract with one or more reciprocal contracts by other Owners, contracts which offer the reverse conversion at a MinRate which allows matching of the contracts.

Each clause has a MinRate value, which is the minimum rate at which the conversion can occur. On the initial execution of a new Backer Change clause, the actual rate at which it occurs is higher than MinRate, if the **Executor** can find a matching contract going the other way that allows for the **Executor** 's fee. Partial conversion is supported, as well as conversions by matching multiple reciprocal contracts.
After the initial execution, if there is any amount left to be converted, the contract goes into the outstanding offers list. If the contract is matched with a new incoming contract before it expires, it receives exactly the MinRate.

The Backer Change Contract cannot be shown using Fig.

15, since Fig. 15 only shows one Backer for all amounts.

Therefore, go back to the initial state of Fig. 15 and assume a new party Y acting as Backer for a new accounting unit, CyberYen. Y authorizes Union Bank to hold accounts backed by Y and Union Bank offers a service converting Y units into $ units and vice versa, all performed by the SAXAS server.

51
If Owner Ted wants to convert $5 from his Union Bank account into Y at a minimum rate of 100Y per $, this can be notated as a Backer Change Offer in SAXAS:

(Union,Ted,$) $5 @10 4 (Union,Ted,Y)
If the Union SAXAS server can complete this Backer Change and sends the notification copy to Ted, the followin transfers occur:

On Union Server
(Union,Ted,$) $10 - $5 $5 Extract $5 from Ted's account (Union,Ted,Y) YO + Y500 Y500 Credit new Y account for Ted On Ted's Server (Ted,Ted,$) $10 - $5 = $5 Reduce total $ owned by Ted (Ted,Union,$) -$10+$5 = -$5 Reduce $ owed by Union to Ted (Ted,Ted,Y) YO + Y500 Y500 Total Y units owned by Ted (Ted,U,Y) YO - Y500 -Y500 Y units owed to Ted by Union However, where did the Y500 come from and where did the $5 go to? This Backer Change can only occur if there is someone else with a contract on the Union server offering to convert 500Y to $ at a rate of 0.01 Y per $. Assume that Bill is a party with Y1000 and $0 and the following SAXAS contract offer. Then the Union server can match up Bill's contract with Ted's contract and make the conversion:

52
(Union,Bill,Y) Y500 @0.01 -> (Union,Bill,$) On Union Server
(Union,Bill,Y) Y1000-Y500=Y500 Extract Y500 from Bill's acct

(Union,Bill,$) $0 + $5 = $5 Credit new account $5 for Bill On Bill's Server

(Bill,Bill,Y) Y1000-Y500=Y500 Reduce total Y owned by Bill (Bill,Union,Y) -YIOOO+Y500=-Y500 Adjust Bill's Y at Union (Bill,Bill,$) $0 + $5 = $5 Total $ owned by Bill (Bill,Union,$) $0 - $5 = -$5 $ owed by Union to Bill As a result of these Backer Changes, the total Y and $ held by Union Bank has not changed, but they are allocated to different Owner accounts. In order for the Union server to stay in balance, both of these Backer Changes must be executed at the same time as a single unit, even though there are two contracts.

Fig. 19a shows an example of two reciprocal backer change clauses that will be matched and executed. Below are two contracts that can be matched up and executed, if the fee is zero. Assume Holder A, Owners B and C, Backers D and E: ABD => ABE 100 @ MinRate 0.5 (100D * 0.5 = 50 E) ACE => ACD 100 @ MinRate 2.0 (50E * 2.0 = 100 D) 53 This is interpreted as "Owner B offers to exchange up to 100 units of D for 0.50 units of E per D, or better. And Owner C offers to exchange up to 50 units of D, the price being 2.0 E per D or less". The **executor** can match these two contracts, exchanging 50 units of E from owner C for 100 units of D from owner B.

In the example of Fig. 19a, the reciprocal backer clauses are at exactly the same price. This does not leave any spread for the **executor** to take a fee.
Fig. 19b shows a different situation where there is a spread between the bid and offer.

ABD => ABE 100 @ MinRate 0.48 100D * 0.48 = 48 E ACE => ACD 100 @ MinRate 2.00 50E * 2.00 = 100 D The Fig. 19b example is the same as Fig. 19a, except that owner B has lowered his price. He will part with 100 D units for 48 E units instead of 50 E. Since owner C will give up 50 E units for 100 D, there are 2.0 units of E left over, from which the **executor** can extract his fee.

Assume that the **executor** fee in Fig. 19b is only 1.0 E.

The remaining 1.0E of spread goes to the new contract. If we assume that B's contract is new and C's is already in the system waiting for new offers, B would get 49E for 100D instead of 48E.

The fee that the **Executor** will charge must be stored in the database somewhere so the software can match the trades automatically. They are available to the outside world via a Name Service request.

54
Fig. 20-21 shows waiting bids and offers and how they are matched with a new, incoming contract. Note that none of the existing Backer change clauses can be matched because the sellers are asking more than the buyers are willing to pay. So the clauses wait on the Holder's server for a new Backer change clause to appear.

If the new contract falls in the price gap between the unmatched clauses, it will be matched with one or more of them.

The new offer to sell 60 units of D at a MinRate of 4.25 E per D can be matched with the two best offers to sell D for E.

The best waiting offer is to sell 50E at 4.40 E per D, which equals 11.37D, plus a 1% commission of 0.1137D, gives a price of 11.4837D.

11.37D goes to the party selling the 50E.

50E goes into the account of the new contract.

0.1137D goes to the **executor** .

11.4837D comes out of the account of the new contract.
This leaves an outstanding new contract to sell 48.5163D at 4.25E or better.

The effective rate on this exchange was 4.35 E per D to the new contract, or better than the offer of 4.25.

The second best waiting offer is to sell 137E at 4.30 E per D, which matches the amount remaining on the new contract. Therefore this order is matched as well, 55 generating a price of 31.86D plus a 1% commission of 0.3186D, which equals 32.1786D.

31.86D goes to the party selling the 137E.

0 137E goes into the account of the new contract.

0 0.3186D goes to the **Executor** .

0 32.1786D comes from the account of the new contract.

0 The effective rate for this exchange is 4.257, which is better than 4.25.
* There is still an outstanding offer to sell 16.3377D at 4.25 E per D, but there are no more waiting offers to match.

Therefore, the outstanding balance goes into the list of buy and sell orders as the best unmatched sell offer. It will be matched with the first new contract that comes in and offers 4.25. The new contract gets the benefit of the best price. The old contracts that are waiting on the server for further matches will always get their MinRate and no better, but they don't have to pay any fees. The new contract effectively pays the **Executor** 's fee if there is one.

Fig. 21 shows the resulting buy and sell orders after the new contract is processed. Notice the two offers to sell E for D are gone and new offer to sell D for E has been added.

56
Alternative Embodiments
Alternative Execution Models
In the preferred embodiment, a clause in a SAXAS contract changes only one element of the Identity triplet defining an account: either the holder, the owner, or the backer. This simplifies the design and the implementation and makes it less likely that any units will ever be lost. It is also desirable to change two of the elements in a single clause.

Since any arbitrarily complex transfer can be accomplished with combinations of contracts, other embodiments of the SAXAS user interface can allow an owner to transfer an amount from any currency, any place, to any other owner in any other currency at any other place, using the contracts in the preferred embodiment.

Also, the holder change can be performed in a single contract instead of two contracts (holder to backer and backer to holder). The logic is similar to the two contract method, but occurs as a single logical step:

1) Owner signs holder change contract
2) Holder B agrees to hold for that owner 3) Backer agrees to hold for the holder B 4) Holder A checks that owner has the funds and updates his books) Backer checks that holder A has funds, updates books and marks contract as-"executed" and notifies owner, B, and A 57 6) Owner, and holder B update their books The Backer is the **executor** of Holder Change contracts, because the Backer is the only one who can give any assurance that the holder still has the amount that is being transferred. Actually,

the Backer can only verify that the holder has at least the amount in the contract in total funds; the Backer cannot verify that the holder has not done something amiss with other amounts.

Alternate Software Methods
The preferred embodiment of this invention was written in the Java language, but there is no reason why it could not be rewritten in C, C++, COBOL, FORTRAN, Pascal, CodeWarrior for the Palm Pilot, or any other computer language.

Alternate Data Store Methods
The preferred embodiment of this invention used the JDBC (Java Database Connectivity) interface to databases, but the invention could be implemented with any database interface that supports tables and transaction commit, including but not limited to DB-2, Oracle, Sybase, Mini-SQL.

Alternate Packaging
The preferred embodiment of this invention is a standalone program that performs its own communication and display, but it could also be embodied as a browser plug in.

58
Alternate Computing Devices
The preferred embodiment of this invention is an Intel Microsoft personal computer, but the invention can be easily implemented on a wide range of computing platforms, including but not limited to UNIX, IBM AS/400, Hewlett Packard MPE/iX, DEC VAX, and IBM mainframe. A database is not needed; only a way to store and recall contracts. A network connection is not needed, only a way to send a contract from one party to another. The contract could even be printed, mailed, scanned, converted back to characters by optical character recognition, and it would still be valid.

A computer is not strictly needed, only an algorithmic device that can parse the contract, compute the mathematical signatures, and send the contract on in some fashion. Given advances in various technologies, this could be a specialized piece of hardware or a genetically engineered biological entity.

While the invention has been described with reference to specific embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted for elements thereof without departing from the true spirit and scope of the invention.

In addition, modifications may be made without departing from the essential teachings of the invention.

Claim
We claim:

59
1. A method for exchanging ownership of values among two or more parties, comprising the steps of providing a computing device as a means to input, output and store information and perform mathematical and logical algorithms on said information, providing a secure channel for sending information intended for a specific party from said computing device to another computing device, and providing a computer program to control said computing device, said computer program being capable of:

operating on several computing devices under control of native parties that control the computing device, creating a digital identity for said party, said digital identity comprising a unique, verifiable sequence of bits that can create a digital signature for a specific digital document and a plurality of optional attributes, said optional attributes comprising the specific secure channel to be used in sending information to the party, the real world identity of the party, and the exchange services provided by the party, receiving said identities when sent over said secure channel, creating an accounting unit for said value, such accounting unit being divisible into fractional parts, being transferable, and being defined by a backer party who defines terms under which said accounting unit may be surrendered for said value, receiving information regarding said accounting units when sent over said secure channel, creating accounts, said accounts to contain a balance amount, said balance amount to be recorded in one of said accounting units, controlled by a owner party, and entrusted to a holder party, and creating a contract in digital form.

2. The method for exchanging ownership of values among two or more parties as set forth in Claim 1, further including receiving said contracts when sent over said secure channel, and sending said contracts over said secure channel to parties on other said computing devices.

3. The method for exchanging ownership of values among two or more parties as set forth in Claim 2, further including signing said contract using said digital identity of a party native to said computing device, verifying that said contract is validly signed by the digital identities that are named as parties in said contract, executing said exchange clauses in said contracts when said contracts are validly signed by all impacted parties and when the designated **executor** party is native to said computing device.

4. The method for exchanging ownership of values among two or more parties as set forth in Claim 3, wherein said step of executing comprises the steps of moving accounting units from one owner party to another owner party, converting amounts from one accounting unit to another accounting unit, matching and executing reciprocal buy and sell contracts, transferring amounts from one holder party to another holder party via the control of the backer party, adjusting account balances in a double-entry manner so that they balance to zero in total, optionally processing said textual agreement, causing information on said parties, said accounts, or said contracts to be appended to the contract and signed by the **executor** , notifying all parties to the contract by sending a copy over said secure channel, and optionally signalling software programs external to the method that said contract has been executed.

5. The method for exchanging ownership of values among two or more parties as set forth in Claim 4, wherein said contract comprises a header, said header comprising the author party and **executor** party and contract type, said contract types specifying special interpretation of the text agreement, a list of party identities, a list of exchange clauses, said text agreement comprising a legal agreement among the parties and, optionally, directions to be interpreted by the software program for specific contract types, and a list of digital signatures by said parties.

6. The method for exchanging ownership of values among two or more parties as set forth in Claim 5 further including creating said exchange clauses, said exchange clauses comprising a route for the value exchange, said route comprising a from account and a to account, an amount, a debit amount indicator if said amount is to be removed from said from account or a credit amount indicator if said amount is to be added to said to account, a minimum conversion rate, and a mortality field to determine how long the clause should remain executable.

7. A secure architecture for performing and managing value exchanges among parties comprising:

a formalized legal contract in a special format, a public key that identifies each party to the value exchange, a software program for use on a programmable computer for each party that can create, validate, sign, store, encrypt, transmit, and execute contracts, a communication link between said computers for transmitting the value exchange information, and a database for each party to hold the contracts and identify information for each party.

8. The secure architecture of Claim 7 wherein said public key is also used to encrypt the transmitted value exchange information to ensure that only the intended party can read the message transmitted.

9. The secure architecture of Claim 8 wherein each party is identified by a unique identity, said identity including said public key.

63

10. The secure architecture of Claim 9 wherein said unique identity includes at least one of a party's preferred name, telephone number, Internet web page address, and system services offered.

11. The secure architecture of Claim 7 further including a personal computer upon which said software program is installed.

12. The secure architecture of Claim 11 wherein said personal computer's operating system is Windows 95 or Windows 98.
13. The secure architecture of Claim 12 wherein said operating system also include the JAVA program.

14. The secure architecture of Claim 7 wherein said public key comprises a sequence of unforgeable characters that uniquely and securely identify a particular party.

iS. The secure architecture of Claim 14 wherein said public key further includes a digital certificate, said certificate comprising the real world identity of a particular party, signed and verified by a certifying authority.

16. The secure architecture of Claim 7 wherein each party to the transaction has an account, said account comprising the identities of the parties, said parties including at least a Holder, an Owner, and a Backer.

17. The secure architecture of Claim 16, wherein a Holder is one of a bank, an escrow agent, or a broker.

64

18. The secure architecture of Claim 16, wherein an owner is the signatory for one or more accounting unit balances, said accounting unit comprising one of any legal tender, or other unit of value for which one party would promise to redeem in return for something with a real world value.

19. The secure architecture of Claim 16, wherein a Backer is that party to the transaction that issues an accounting unit, said accounting unit comprising one of any legal tender, or other unit of value for which one party would promise to redeem in return for something with a real world value.

20. The secure architecture of Claim 7, wherein said formalized contract comprises at least four electrically signed sections, said sections comprising a header section which includes at least one of a title, author, **executor** , date of contract maturity, a parties section which includes the identification of the party, a clauses section which includes the amount of the contract and the interest rate, an agreement section which includes the actual text of the contract agreement.

21. The secure architecture of Claim 20, further including a fifth

section, said section comprising the electronic signatures of all the parties to the contract.

22. The secure architecture of Claim 12 wherein said operating system also includes at least one of the following programs: C, C++, COBOL, FORTRAN, Pascal, CodeWarrior.

23. The secure architecture of Claim 7 further including a database interface, said database interface including at least one of DB-2, Oracle, Sybase, Mini-SQL.

24. The secure architecture of Claim 11 wherein said personal computer is one of the following computers: Intel Microsoft family of computers, UNIX, IBM AS/400, Hewlett Packard MPE/iX, DEC VAX, IBM mainframe.

?